Fundamenta Informaticae 119 (2012) 1–27
DOI 10.3233/FI-2012-670
IOS Press

Enumeration and Leader Election in Partially Anonymous and Multi-hop Broadcast Networks

Jérémie Chalopin
Université Aix-Marseille, LIF, CNRS
39, rue Joliot Curie, 13453 Marseille Cedex 13, France
jeremie.chalopin@lif.univ-mrs.fr

Yves Métivier, Thomas Morsellino[*]
Université de Bordeaux, LaBRI, UMR CNRS 5800
351, cours de la Libération, 33405 Talence, France
{*metivier, morsellino*}*@labri.fr*

Abstract. We address the enumeration and the leader election problems over partially anonymous and multi-hop broadcast networks. We consider an asynchronous communication model where each process broadcasts a message and all its neighbours receive this message after arbitrary and unpredictable time. In this paper, we present necessary conditions that must be satisfied by any graph to solve these problems and we show that these conditions are sufficient by providing an enumeration algorithm on the one hand and a leader election algorithm on the other hand. For both problems, we highlight the importance of the initial knowledge. Considering the enumeration problem, each process only knows the size of the graph and, contrary to related works, the number of its neighbouring processes is unknown. Whereas for the election problem, we show that this combination of knowledge is not sufficient. Our algorithm assumes that each process initially knows a map of the network (without knowing its position in this map). From the complexity viewpoint, our algorithms offer polynomial complexities (memory at each process, number and size of exchanged messages).

Keywords: Distributed Computing, Multi-hop, Broadcast, Election, Naming, Enumeration, Fibrations

[*]Address for correspondence: Université de Bordeaux, LaBRI, UMR CNRS 5800, 351, cours de la Libération, 33405 Talence, France

1. Introduction

A multi-hop and ad-hoc broadcast network is a collection of processes which communicate by broadcasting messages and should run in absence of any preexisting infrastructure (e.g., ad-hoc wireless networks). Some of the important challenges in such a network are enumeration and leader election which are well-known in the field of distributed systems [14, 17, 18, 19, 22].

1.1. Enumeration and Election

The aim of a naming algorithm is to give pairwise distinct identities to all processes. The enumeration problem is a variant of the naming problem and aims at giving to each process a unique number between 1 and the size of the graph. Existence of identified processes allows better routing of information, resource management and performance [19].

A distributed algorithm solves the election problem [15] if it always terminates and in the final configuration exactly one process is marked as *elected* and all the other processes are marked as *non-elected*. Moreover, it is supposed that once a process becomes *elected* or *non-elected* then it remains in such a state until the end of the algorithm. Election algorithms constitute a building block of many other distributed algorithms. The elected vertex acts as coordinator, initiator, and more generally performs some special role (see [21] p. 262).

Using enumeration/naming algorithm, one can promote the process with the highest (resp. lowest) identifier as *elected*. However, enumeration and election problems are not necessarily equivalent (see [5, 9]). We are here interested in characterizing graphs for which there exists an algorithm that solves the enumeration and the naming problems or that solves the election problem.

1.2. The Model

We consider an asynchronous broadcast communication model (see [11, 10]). A network is represented by a simple connected graph $G = (V(G), E(G)) = (V, E)$ where vertices correspond to processes and edges to direct communication links. The state of each process is represented by a label $\lambda(v)$ associated to the corresponding vertex $v \in V(G)$; we denote by $\mathbf{G} = (G, \lambda)$ such a labelled graph.

Remark 1.1. Labels (states) are attached to vertices. They make it possible to encode many different situations. If the network is *anonymous* then all vertices have the same label; vertices having unique identities, a distinguished vertex or any intermediate situation, qualified as *partially anonymous*, are other examples of labels attached to vertices.

We consider a robust model in which we assume partially anonymous graphs, i.e., processes have names which are not necessarily distinct. The question of anonymity is often considered when processes must not divulge their identities during executions, due to privacy concerns or security policy issues [13]. In addition, each process may be built in large scale quantities from which it is quite infeasible to ensure uniqueness. Therefore, each process must execute the same finite algorithm in the same way, regardless of its identity (see [1, 2] for related works on anonymity).

Emitted messages are only heard by reachable processes. We consider ad-hoc networks which run in absence of any infrastructure and relying on the message passing model and asynchronous broadcast communications: processes cannot access a global clock and execute computation steps (atomic emit, hear and internal computation) at arbitrary speed. Communication links are reliable but asynchronous, i.e., a message emitted from a process to neighbours arrives within some finite but unpredictable time. Note that communications are not necessarily FIFO.

1.3. Overview of our Contributions

We give complete characterizations of multi-hop broadcast networks where there exists an enumeration algorithm or an election algorithm (Theorem 3.8 and Theorem 4.9). In this model, enumeration and election problems are not equivalent, meaning that even if we can elect a leader, we cannot always give a unique number to every process.

Let $\mathbf{G} = (G, \lambda)$ be a labelled graph. We will denote by $\mathrm{Dir}(\mathbf{G})$ the symmetric labelled directed graph (digraph) $(\mathrm{Dir}(G), \lambda)$ constructed in the following way. The vertices of $\mathrm{Dir}(G)$ are the vertices of G and they have the same labels in $\mathbf{G}$ and in $\mathrm{Dir}(\mathbf{G})$. Each edge $\{u, v\}$ of G is replaced in $\mathrm{Dir}(\mathbf{G})$ by two arcs $a_{(u,v)}, a_{(v,u)} \in A(\mathrm{Dir}(G))$ such that $s(a_{(u,v)}) = t(a_{(v,u)}) = u$, $t(a_{(u,v)}) = s(a_{(v,u)}) = v$. Note that this digraph does not contain multiple arcs or loop. The object we use for our study is $(\mathrm{Dir}(G), \lambda)$ and results are stated with symmetric labelled digraphs.

A fibration from a digraph $\mathbf{D}$ to a digraph $\mathbf{D}'$ is a homomorphism from $\mathbf{D}$ to $\mathbf{D}'$ that induces an isomorphism between the incoming arcs of each vertex of D and the incoming arcs of its image.

First, we prove that, in the asynchronous broadcast model, there exists an enumeration algorithm if and only if $\mathrm{Dir}(\mathbf{G})$ is minimal for the fibration relation, i.e., if there exists a fibration from $\mathrm{Dir}(\mathbf{G})$ to $\mathbf{D}'$ then it is an isomorphism.

For the election problem, we prove that there exists an election algorithm if and only if once there exists a fibration φ from $\mathrm{Dir}(\mathbf{G})$ to $\mathbf{D}'$ then necessarily there exists a vertex v of D' such that $\varphi^{-1}(v)$ is a singleton.

For both problems, our algorithms do not require each process to know its degree. For the enumeration problem, processes only know the size of the network. However, we show that this initial knowledge is not sufficient when one considers the election problem. Thus, our leader election algorithm assumes that each process knows a map of the network but is not aware of its position in this map.

Furthermore, our algorithms have a polynomial complexity: local memory, number of messages and size of messages are polynomially bounded by the size of the network.

Remark 1.2. (Initial Knowledge)
For the enumeration algorithm, it suffices that every process knows the size of the network for the termination detection. This hypothesis is classical when considering leader election and naming/enumeration problems [1, 5, 23, 25, 17].

For the election algorithm, to detect its termination we assume that each process knows a map of the whole graph (see Section 4.2 for a discussion); we also prove that it suffices that every vertex knows the size of the graph and the size of its neighbourhood (see Section 4.6).

1.4. Related Works: Comparison and Comments

Graphs where election or naming are possible were already studied for different basic models. Solutions depend on the type of basic computation steps, on the type of network topology and on the initial knowledge.

Angluin [1] has introduced the classical proof techniques used for showing the non-existence of an election algorithm based on coverings, which is a notion known from algebraic topology [16]. Finally, several characterizations of graphs for which there exists an election algorithm have been obtained [5, 23, 25, 17].

The model studied in this paper corresponds to the Broadcast-to-Mailbox communication mode of Yamashita and Kameda [25] and to the no output port awareness and no input port awareness of Boldi *et al.* [5]. We use intensively fibrations introduced in [5] and studied in [6]. The fundamental tool in [25, 5] is the notion of view. The view from a vertex v of a labelled graph (G, λ) is an infinite labelled tree rooted in v obtained by considering all labelled walks in (G, λ) starting from v.

The characterization of graphs where election is possible obtained in [25] is formulated by using views whereas Boldi *et al.* [5] use fibrations. In both cases election algorithms are based on views and the election algorithms presented in [25, 5] use messages with an exponential size, they need the knowledge of the size of the graph and the size of the neighbourhood of each vertex; this knowledge is used in the algorithms to ensure that all executions are pseudo-synchronous and that communication links behave like FIFO channels.

Techniques developed in this paper are inspired by the work of Mazurkiewicz [17]. He considers the asynchronous computation model where in one computation step labels of vertices are modified on a subgraph consisting of a process and its neighbours, according to rules depending on this subgraph only. Mazurkiewicz's characterization of the graphs where enumeration/election are possible is based on the notion of unambiguous graphs and may be formulated equivalently using coverings of simple graphs (see [12], p. 256). A graph G is a covering of another graph G' if there is a surjective homomorphism φ from G to G' which is locally bijective. He gives a nice and simple enumeration algorithm for the graphs that are minimal for the covering relation, i.e., which can cover only themselves. The fundamental tool is a total order attached to local views defined by a vertex and its neighbourhood.

These techniques have been also used in [7, 8]. The model of [7] (it is the same one as [23]) is such that in each step, one of the vertices, depending on its current label, either changes its state, or sends/receives a message via one of its ports. The model of [8] is defined by local computations on labelled edges of graphs. In both cases the election problem and the enumeration problem are equivalent.

Cidon and Mokryn present in [11] an election algorithm in multi-hop radio networks. This algorithm partitions the network into fragments that are collections of processes where one process is identified as a candidate and marked initially as active. They consider networks that are not anonymous: each vertex has a unique identity. During the computation, a candidate can become inactive and joins another candidate's fragment.

1.5. Summary

First, we present in Section 2 the notion of fibration for digraphs and the fundamental lemma (Lemma 2.11) which connects fibrations and asynchronous broadcast communications. In Section 3, we characterize graphs which admit and enumeration algorithm while in Section 4, we characterize graphs which admit an election algorithm.

2. Preliminaries

In order to describe our characterizations, one needs to consider directed graphs (digraphs for short) that can have multiple arcs and self-loops. In this section, we present various definitions about digraphs and labelled digraphs. We also present fibrations which are a particular type of homomorphism. From these definitions, we give a fundamental lemma that establishes a link between fibrations and asynchronous broadcast communications.

2.1. Labelled Simple Graphs and Digraphs

Undirected graphs without multiple edges or loop are also called simple graphs. Each such a graph is written as $G = (V(G), E(G))$ where $V(G)$ is the set of vertices of G and where the set of edges $E(G)$ is a set of pairs of distinct vertices of G. For each edge $\{u, v\} \in E(G)$, u and v are the *ends* of $\{u, v\}$ and u and v are said to be *adjacent* or *neighbours*. We denote by $N_G(u)$ the set of all vertices of G adjacent to u and $\deg_G(u)$ is the degree of u in G, i.e., the size of $N_G(u)$.

A simple graph G is *connected* if for all vertices $u, v \in V(G)$, there exists a path between u and v. Otherwise, it is *disconnected*. In the following, we will only consider connected simple graphs.

A *digraph* with multiple arcs, also called *directed multigraph*, $D = (V(D), A(D), s_D, t_D)$ is defined by a set $V(D)$ of vertices, a set $A(D)$ of arcs and by two maps s_D and t_D that assign to each arc two elements of $V(D)$: a source and a target (in general, the subscripts will be omitted). If a is an arc, the arc a is said to be going out of $s(a)$ and coming into $t(a)$; we also say that $s(a)$ and $t(a)$ are incident to a. Let a be an arc, if $s(a) = u$ and $t(a) = v$ then v is an outgoing neighbour of u and u is an incoming neighbour of v. A self-loop is an arc with the same source and target.

Remark 2.1. Note that since we consider digraphs with multiple arcs, an arc a is not uniquely defined by $s(a)$ and $t(a)$.

A *symmetric* digraph D is a digraph endowed with a symmetry, that is, an involution $Sym : A(D) \to A(D)$ such that for every $a \in A(D), s(a) = t(Sym(a))$. In a symmetric digraph D, the degree of a vertex v is $\deg_D(v) = |\{a \mid s(a) = v\}| = |\{a \mid t(a) = v\}|$ and we denote by $N_D(v)$ the set of neighbours of v which is equal to the set of out-neighbours of v and to the set of in-neighbours of v.

Given two vertices $u, v \in V(D)$, a *path* π of *l*ength p from u to v in D is a sequence of arcs $a_1, a_2, \dots a_p$ such that $s(a_1) = u, \forall i \in [1, p-1], t(a_i) = s(a_{i+1})$ and $t(a_p) = v$. If for each $i \in [1, p-1]$, $a_{i+1} \neq Sym(a_i)$, π is *non-stuttering*. A digraph D is *strongly connected* if for all vertices $u, v \in V(D)$, there exists a path from u to v in D. In a digraph D, the *distance* between two vertices u and v, denoted $dist_D(u, v)$, is the length of the shortest path from u to v in D. Note that $dist_D(u, v)$ is not necessarily equal to $dist_D(v, u)$ unless D is a symmetric digraph. A digraph H is a *subdigraph* of D, noted $H \subseteq D$, if $V(H) \subseteq V(D)$ and $A(H) \subseteq A(D)$.

Definition 2.2. ([3])
A *homomorphism* φ from the digraph D to the digraph D' denoted $\varphi\colon D \to D'$ is a mapping $\varphi\colon V(D) \cup A(D) \to V(D') \cup A(D')$ such that for every vertex $v \in V(D)$, $\varphi(v) \in V(D')$ and for every arc $a \in A(D)$, $\varphi(a) \in A(D')$, $\varphi(s(a)) = s(\varphi(a))$ and $\varphi(t(a)) = t(\varphi(a))$.

A homomorphism φ is an *isomorphism* if φ is bijective. We write $D \approx D'$ whenever D and D' are isomorphic.

In this paper, we consider digraphs where the vertices are labelled with labels from a recursive set L. A digraph D labelled over L will be denoted by (D, λ), where $\lambda\colon V(D) \to L$ is the labelling function. The digraph D is called the underlying digraph and the mapping λ is a labelling of D. A mapping $\varphi\colon V(D) \to V(D')$ is a homomorphism from (D, λ) to (D', λ') if φ is a digraph homomorphism from D to D' which preserves the labelling, i.e., such that $\lambda'(\varphi(v)) = \lambda(v)$ for every $v \in V(D)$. Labelled digraphs will be designated by bold letters like $\mathbf{D}, \mathbf{D}', \ldots$ If $\mathbf{D}$ is a labelled digraph, then D denotes the underlying digraph.

Let H be a subgraph of D and λ_H the restriction of a labelling $\lambda\colon V(D) \to L$ to $V(H)$. Then the labelled graph $H = (H, \lambda_H)$ is called a *subdigraph* of $G = (D, \lambda)$; we note this fact by $\mathbf{H} \subseteq \mathbf{D}$.

Our proofs use the notion of *view*. Informally, the view of a vertex v in a digraph $\mathbf{D}$ is obtained by considering all labelled paths in $\mathbf{D}$ ending in v. From the computation viewpoint, the view of a process in a network is a tree representing all the information it can gather about the network.

Definition 2.3. Given a labelled digraph $\mathbf{D}$, the view $T_{\mathbf{D}}(v_0)$ of a vertex v_0 is an infinite rooted labelled tree that can be defined recursively. The root of the tree is a vertex x_0 that corresponds to v_0 and is labelled by $\lambda(v_0)$. For each incoming neighbour v_i of v_0 in $\mathbf{D}$, there is an arc between x_0 and the root x_i of the tree $T_{\mathbf{D}}(v_i)$. Let d be an integer, the d-view $T^d_{\mathbf{D}}(v_0)$ of $v_0 \in V(D)$ is the infinite view $T_{\mathbf{D}}(v_0)$ truncated at depth d.

From this definition, we can state that the set of d-views of a digraph $\mathbf{D}$ is finite. Thus, we can define a partial order $\succeq$ on this set as follows:

Definition 2.4. For every vertex $v, w \in V(D)$, if $T = T^d_{\mathbf{D}}(w)$ is a subtree of $T' = T^d_{\mathbf{D}}(v)$ then $T' \succeq T$. Note that if there exists an isomorphism between T to T', they are said to be similar, denoted $T \approx T'$.

Remark 2.5. The labelling λ of vertices may encode some properties of the network or an initial knowledge. For example, if the network is anonymous, all the vertices have the same label (i.e., $\forall u, u' \in V(G), \lambda(u) = \lambda(u')$). If the processes have unique identities, then for all $u, u' \in V(G)$ if $u \neq u'$ then $\lambda(u) \neq \lambda(u')$. If there exists a distinguished process, then there exists $u \in V(G)$ such that for each $u' \in V(G)$ distinct from u, $\lambda(u) \neq \lambda(u')$. It may also encode partial identities of processes. As initial knowledge, label of a vertex may encode its degree or the size of the graph.

Remark 2.6. Note that computing the view of a process belongs to the set of tools which allows to capture "symmetric" behaviour in distributed computations. The algorithms of Boldi *et al.* [5] and of Yamashita and Kameda [25] are based on the notion of view.

2.2. Homomorphism and Fibration

Fibrations, t-fibrations and nt-fibrations are important tools for this work (see [4, 6] for definitions and properties).

A fibration is a homomorphism that induces an isomorphism between the incoming arcs of a vertex and the incoming arcs of its image.

Definition 2.7. A digraph D is *fibred over* a digraph D' via a homomorphism φ if φ is a homomorphism from D to D' such that for each arc $a' \in A(D')$ and for each vertex $v \in \varphi^{-1}(t(a'))$, there exists a unique arc $a \in A(D)$ such that $t(a) = v$ and $\varphi(a) = a'$; this arc a is called the *lifting* of a' at v.

We say that the homomorphism φ is a *fibration* from D to D', the digraph D is the *total digraph* of φ and the digraph D' is the *base* of φ.

The *fibre* over a vertex v' (resp. an arc a') of D' is defined as the set $\varphi^{-1}(v')$ of vertices of D (resp. the set $\varphi^{-1}(a')$ of arcs of D).

The digraph $\mathbf{D}$ is *minimal* if for every digraph $\mathbf{D}'$ such that $\mathbf{D}$ is fibred over $\mathbf{D}'$, $\mathbf{D}$ and $\mathbf{D}'$ are isomorphic.

If a digraph $\mathbf{D}$ is fibred over a digraph $\mathbf{D}'$ via a homomorphism φ, and if $\mathbf{D}$ and $\mathbf{D}'$ are not isomorphic, we say that $\mathbf{D}$ is *properly* fibred over $\mathbf{D}'$ and that φ is a *proper* fibration.

From [6], we know that there exists a unique digraph $\mathbf{B_G}$ such that $Dir(\mathbf{G})$ is fibred over $\mathbf{B_G}$, and for each $\mathbf{D}$ such that $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$, $\mathbf{D}$ is fibred over $\mathbf{B_G}$. This digraph is called the *minimal base* of $\mathbf{G}$.

In this work, we need to define t-fibrations and nt-fibrations.

Definition 2.8. The fibre of a vertex v is qualified as *trivial* if $|\varphi^{-1}(v)| = 1$, otherwise, it is *non-trivial*.

A fibration φ is a t-fibration if there exists at least one vertex such that its fibre is trivial; it is a nt-fibration if all fibres are non-trivial.

A digraph D is t-fibred (resp. nt-fibred) over a digraph D' via φ if and only if φ is a t-fibration (resp. nt-fibration).

The digraph $\mathbf{D}$ is nt-minimal if for every digraph $\mathbf{D}'$ such that $\mathbf{D}$ is fibred over $\mathbf{D}'$ via a fibration φ, φ is a t-fibration.

A simple graph $\mathbf{G}$ is minimal if $Dir(G)$ is minimal. Similarly, a simple graph $\mathbf{G}$ is nt-minimal if $Dir(\mathbf{G})$ is nt-minimal. An example of fibration is given in Figure 1.

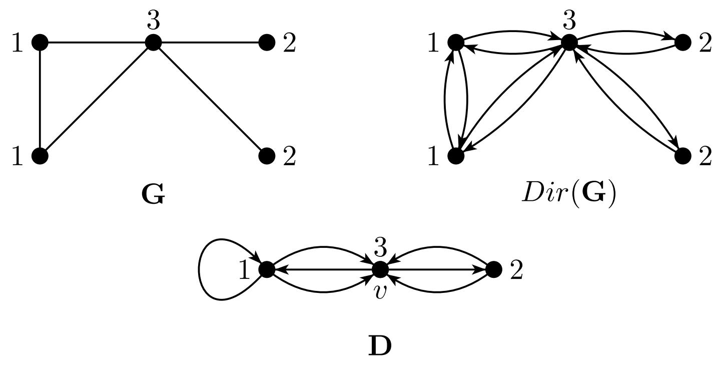

Figure 1. The labelled digraph $Dir(\mathbf{G})$ is fibred over the digraph $\mathbf{D}$. Therefore, $Dir(\mathbf{G})$ is not minimal. Since $Dir(\mathbf{G})$ has a unique vertex of degree 4, $Dir(\mathbf{G})$ is nt-minimal. The digraph $\mathbf{D}$ is minimal and also nt-minimal.

Remark 2.9. As a corollary of Definition 2.4, we obtain: let $\mathbf{H}$ be a sub-digraph of $Dir(\mathbf{G})$, for every vertex $v \in Dir(\mathbf{G})$, $T^d_{Dir(\mathbf{G})}(v) \succeq T^d_{\mathbf{H}}(v)$.

Moreover, let $\mathbf{D}$ and $\mathbf{D}'$ be two digraphs. If $\mathbf{D}$ is fibred over $\mathbf{D}'$ via φ, then $T_{\mathbf{D}}(v) \approx T_{\mathbf{D}'}(\varphi(v))$, i.e, the view of v in $\mathbf{D}$ is isomorphic to the view of $\varphi(v)$ in $\mathbf{D}'$.

Note that the vertices of the minimal base $\mathbf{B}$ of $\mathbf{G}$ can be identified to their views in $\mathbf{B}$: this defines a unique homomorphism from $\mathbf{G}$ to $\mathbf{B}$. We define the notion of candidate for a digraph $\mathbf{D}$ such that $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$.

Definition 2.10. Consider a $\mathcal{nt}$-minimal graph $\mathbf{G}$, let $\mathbf{B}$ be the minimal base of $Dir(\mathbf{G})$, and let φ be the unique fibration from $Dir(\mathbf{G})$ to $\mathbf{B}$. A vertex $v \in V(B)$ is a candidate of $\mathbf{B}$ if $|\varphi^{-1}(v)| = 1$, i.e., if there is a unique vertex $w \in V(G)$ such that $\mathbf{T_G}(w) \approx \mathbf{T_B}(v)$.

Given a digraph $\mathbf{D}$ such that $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$, we know that $\mathbf{D}$ is fibred over $\mathbf{B}$ via a unique homomorphism φ'. A vertex v is a candidate of $\mathbf{D}$ if and only if $\varphi'(v)$ is a candidate of $\mathbf{B}$.

We denote by $C_{\mathbf{G},\mathbf{D}}$ the set of candidates of $\mathbf{D}$.

Note that if a $\mathcal{nt}$-minimal digraph $Dir(\mathbf{G})$ is fibred over a digraph $\mathbf{D}$ via a homomorphism φ, then for every vertex $v \in C_{\mathbf{G},\mathbf{D}}$, $|\varphi^{-1}(v)| = 1$.

Intuitively, a leader election algorithm on a graph $\mathbf{G}$ fibred over $\mathbf{D}$ cannot declare a vertex which does not belong to the set of candidate $C_{\mathbf{G},\mathbf{D}}$ (see Section 4).

2.3. Fibrations and Broadcast Communications

In order to extend the Lifting Lemma of Angluin [1] and Boldi *et al.* [5] to asynchronous broadcast communications, we present the correlation between fibrations and asynchronous broadcast communications.

Leader election and enumeration problems require the network to reach a *non-symmetric* state. A network state is qualified as symmetric if it contains different processes that are in exactly the same situation; not only their local states, but also the states of their neighbors, of their neighbors' neighbors, etc. That is, there exists a "local similarity" between different processes of infinite radius.

The replay argument shows that different processes that are locally similar with infinite radius will exhibit the same behaviour in some infinite computation. Thus, there is no algorithm that guarantees that the symmetry ceases in all finite computations.

It is not difficult to see that local similarity of infinite radius may exist in finite graphs. It is precisely captured by the notion of graph coverings used by Angluin and this is the mathematical tool to prove the existence of symmetries of infinite radius.

In our model, when a process emits a message, it modifies its state according to only its previous state, while its neighbouring processes that hear the message modify their states following their previous states and the state of the emitting process.

Thus, multi-hop broadcast networks in which symmetries exist are non minimal and impossibility of symmetry breaking can be shown for these graphs. The following lemma connects fibrations and asynchronous broadcast communication steps.

A *maximal* execution ρ of an algorithm is either an infinite execution, or a finite execution such that in the final configuration, there is no message in transit and no process wants to emit a message.

Lemma 2.11. (Asynchronous Lifting Lemma)
Consider a digraph $\mathbf{D}_1$ fibred over a digraph $\mathbf{D}_2$ via φ and let $\mathcal{A}$ be an algorithm based on the asynchronous broadcast model. If there exists a maximal execution ρ_2 of $\mathcal{A}$ on $\mathbf{D}_2$ which yields $\mathbf{D}'_2$ then there exists a maximal execution ρ_1 of $\mathcal{A}$ on $\mathbf{D}_1$ which yields $\mathbf{D}'_1$ such that $\mathbf{D}'_1$ is fibred over $\mathbf{D}'_2$ via φ.

Proof:
Let $\mathbf{D}_1 = (D_1, \lambda_1), \mathbf{D}_2 = (D_2, \lambda_1)$ be two digraphs such that (D_1, λ_1) is fibred over (D_2, λ_2) via φ.

Consider a particular set of executions Π on $\mathbf{D}_2$ in which each emitted message from a process v is followed by the hearing of all its neighbours. Consider a step of $\rho \in \Pi$: the process v emits a message in $\mathbf{D}_2$ and all its neighbours hear the message just after its emission. Let λ_2' be the labelling of $\mathbf{D}_2$ after this step. One can lift this execution in $\mathbf{D}_1$ in which every vertex in $\varphi^{-1}(v)$ emits the same message (not simultaneously and in any order). Then, all emitted messages are heard. Let denote λ_1', the new labelling of $\mathbf{D}_1$. Each vertex $w \in N_{\mathbf{D}_2}(v)$ hears k messages, with k depending on the number of arcs $a \in A(\mathbf{D}_2)$ such that $s(a) = v$ and $t(a) = w$. Since φ is a fibration relation, for every vertex $w' \in \varphi^{-1}(w)$, w' has k neighbouring processes in $\varphi^{-1}(v)$ and hears k same messages. In this sense, $\lambda_1'(w') = \lambda_2'(w)$ and labels of all other vertices are not modified. Note that if there exists any self-loop on v, then there exist arcs $a \in A(\mathbf{D}_2)$ such that $s(a) = t(a) = v$. Once v has emitted a message, $\lambda_1'(v) = \lambda_2'(\varphi^{-1}(v))$. Thereafter once v has heard this message, we have also $\lambda_1'(v) = \lambda_2'(\varphi^{-1}(v))$. Therefore, the digraph (D_1, λ_1') is fibred over (D_2, λ_2') via φ. Thus, if the execution ρ is infinite on $\mathbf{D}_2$, the lifted execution on $\mathbf{D}_1$ is also infinite. If the maximal execution ρ on $\mathbf{D}_2$ is finite, then all messages have arrived, and no process has to emit a message. Hence, after the execution lifted form ρ on $\mathbf{D}_1$, $\mathbf{D}_1$ is fibred over $\mathbf{D}_2$ and all messages have also arrived and no process has to emit a message: the lifted execution is maximal. □

3. An Enumeration Algorithm for Broadcast Networks

In this section, we give a necessary condition based on an impossibility result which states that there exists no enumeration algorithm for a graph $\mathbf{G}$ such that $Dir(\mathbf{G})$ is not minimal. Then, we prove that this condition is sufficient by presenting an enumeration algorithm (Algorithm 1) which relies on asynchronous broadcast communications and is inspired by the work of Mazurkiewicz [17].

3.1. Impossibility Result

Given a network represented by a graph $\mathbf{G}$, we present a necessary condition that must be satisfied by $\mathbf{G}$ to admit an enumeration algorithm. This is an impossibility result that relies on the notion of fibrations for asynchronous computations. Following the proof of Lemma 2.11 presented above, we show that two processes belonging to a same fibre cannot have different names.

Proposition 3.1. Let $\mathbf{G}$ be a labelled graph such that $Dir(\mathbf{G})$ is not minimal, there is no enumeration algorithm for $\mathbf{G}$ in the asynchronous broadcast model.

Proof:
Consider a simple graph $\mathbf{G} = (G, \lambda)$ and a strongly connected digraph $\mathbf{D} = (D, \eta)$ such that $Dir(\mathbf{G})$ is properly fibred over $\mathbf{D}$ via a fibration φ. Given an algorithm $\mathcal{A}$ relying on asynchronous broadcast communications, consider an execution of $\mathcal{A}$ on $\mathbf{D}$ as described in Lemma 2.11. Note that if this execution of $\mathcal{A}$ on $\mathbf{D}$ is infinite, then following Lemma 2.11 there exists an infinite execution of $\mathcal{A}$ on $\mathbf{G}$. Finally, $\mathcal{A}$ is not an enumeration algorithm for $\mathbf{G}$.

Suppose this execution of $\mathcal{A}$ on $\mathbf{D}$ is finite and yields a configuration $\mathbf{D}'$. In the final configuration every message has arrived and no process has to emit a message. Thus, each vertex has its final label. Following Lemma 2.11, there exists a lifted execution of $\mathcal{A}$ on $Dir(\mathbf{G})$ that yields a configuration $\mathbf{G}'$

such that $\mathbf{G}'$ is properly fibred over $\mathbf{D}'$ via φ. Since $\mathbf{G}'$ is fibred over $\mathbf{D}'$ it implies that there exist at least two vertices that have the same label in $\mathbf{G}'$. Hence, the algorithm $\mathcal{A}$ does not give a distinct label to each vertex and is not an enumeration algorithm for $\mathbf{G}$. □

3.2. Informal Description of the Enumeration Algorithm

We first give a general description of our algorithm, that will be denoted $\mathcal{M}$, when executed on a connected labelled simple graph $\mathbf{G}$.

During the execution of the enumeration algorithm, each vertex v attempts to get its unique identity label: a number between 1 and $|V(G)|$. Once a vertex v has chosen a number $n(v)$, it emits it to its neighbourhood. When a vertex v hears a message from a neighbour u, it stores the number $n(u)$. From all information it has gathered from its neighbours, each vertex v is able to create its *local view*. Schematically, the local view of v is the multiset of given numbers that appear in his neighborhood. Then, a vertex broadcasts its number with its *local view* $N(v)$. If a vertex u discovers that there exists another vertex v with the same number then it should decide if it changes its identity: it compares its local view with the local view of v. If the label of u or the local view of u is *weaker* (for an order we define later), then u chooses another identity and emits it again with its local view. At the end of the computation, if the digraph $Dir(\mathbf{G})$ is minimal, then every vertex will have a unique number.

3.2.1. Labels

We consider a network $\mathbf{G}$ where $\mathbf{G} = (G, \lambda)$ is a simple labelled graph. The function $\lambda : V(G) \to L$ is the initial vertex labelling and is kept during the computation. We suppose that there exists a total order $<_L$ on L. During the execution, the label of each vertex v is a tuple $(\lambda(v), n(v), N(v), M(v))$ corresponding to the following information:

- $\lambda(v) \in L$ is the initial label of v and is not modified by the algorithm.
- $n(v) \in \mathbb{N}$ is the current *number* of the vertex v computed by the algorithm.
- $N(v) \in \mathcal{P}_{\text{fin}}(\mathbb{N} \times \mathbb{Z})$[1] is the *local view* of v. Intuitively, once v has updated its local view, (n, p) belongs to $N(v)$ if v knows p neighbours that have n as an identity number.
- $M(v) \in \mathbb{N} \times L \times \mathcal{P}_{\text{fin}}(\mathbb{N}^2)$ is the *mailbox* of v. The mailbox of v contains all information heard by v during the execution of the algorithm. If $(m, \ell, \mathcal{N}) \in M(v)$, it means that at some previous step of the execution, there was a vertex u such that $n(u) = m$, $\lambda(u) = \ell$ and $N(u) = \mathcal{N}$.

Initially, each vertex v has a label of the form $(\lambda(v), 0, \emptyset, \emptyset)$ indicating that it has not chosen any number, that it has no information about its neighbours or about the other vertices of the graph.

In order to update the local view of a process $v_0 \in V(G)$, we define a function $update(n, n_{old})$ the operations defined as follows. First, if $n_{old} \neq 0$, we apply the following rule:

- if there exists $(n_{old}, 1) \in N(v_0)$ then $N(v_0) := N(v_0) \setminus \{(n_{old}, 1)\}$,

[1]For any set S, $\mathcal{P}_{\text{fin}}(S)$ denotes the set of finite subsets of S.

- if there exists $(n_{old}, p) \in N(v_0)$ with $p \neq 1$ then $N(v_0) := N(v_0) \setminus \{(n_{old}, p)\} \cup \{(n_{old}, p-1)\}$,
- otherwise, $N(v_0) := N(v_0) \cup \{(n_{old}, -1)\}$.

Then, symmetrically, we do the following operations:

- if there exists $(n, -1) \in N(v_0)$ then $N(v_0) := N(v_0) \setminus \{(n, -1)\}$,
- if there exists $(n, p) \in N(v_0)$ with $p \neq -1$ then $N(v_0) := N(v_0) \setminus \{(n, p)\} \cup \{(n, p+1)\}$,
- otherwise, $N(v_0) := N(v_0) \cup \{(n, 1)\}$.

3.2.2. Messages

In our algorithm, processes exchange messages of the form $< (m, n_{old}, M) >$. If a vertex u emits a message $< (m, n_{old}, M) >$, then m is the current number $n(u)$ of u, n_{old} is the previous number of u; if in the meanwhile, u has not modified its number, then $n_{old} = m$ and M is the mailbox of u.

Remark 3.2. If there exists $(n, p) \in N(v)$ with $p < 0$, then it means that among all the messages $< (m, n_{old}, M) >$ that v has heard, there are more messages where $n_{old} = n$ than messages where $m = n$. However, each time a process w emits a message $< (m, n_{old}, M) >$ with $m \neq n_{old}$, we know that w has previously emitted a message $< (n_{old}, n'_{old}, M) >$ with $n_{old} > n'_{old}$.

Consequently, if there exists $(n, p) \in N(v)$ with $p < 0$, then it implies that v has not heard yet all messages sent by its neighbours, and thus it can wait until it hears a message of the form $< (m, n, M) >$.

3.2.3. An Order on Local Views

As in Mazurkiewicz's algorithm [17], the nice properties of the algorithm rely on a total order on local views, i.e., on finite subsets of $\mathcal{P}_{\text{fin}}(\mathbb{N}^2)$. The algorithm described above is such that the local view of any vertex cannot decrease during the computation.

In order to compare two elements of $\mathbb{N}^2$, we use the usual lexicographic order on $\mathbb{N}^2$: $(n, p) < (n', p')$ if $n < n'$, or if $n = n'$ and $p < p'$.

Let $N_1, N_2 \in \mathcal{P}_{\text{fin}}(\mathbb{N}^2)$, $N_1 \neq N_2$. Consider (n, p) as the maximal element of the symmetric difference $N_1 \triangle N_2 = (N_1 \setminus N_2) \cup (N_2 \setminus N_1)$. Then $N_1 \prec N_2$ if and only if one of the following conditions holds:

- $(n, p) \in N_1$ and $p < 0$,
- $(n, p) \in N_2$ and $p > 0$.

If $N(u) \prec N(v)$ then we say that the local view $N(v)$ of v is *stronger* than the one of u (and $N(u)$ is *weaker* than $N(v)$). Note that in particular the empty set is minimal for $\prec$. We assume for the rest of the paper that the set of initial labels L is totally ordered by $<_L$. We extend $\prec$ to a total order on $L \times \mathcal{P}_{\text{fin}}(L \times \mathbb{N})$: $(\ell, N) \prec (\ell', N')$ if either $\ell <_L \ell'$, or $\ell = \ell'$ and $N \prec N'$. We denote by $\preceq$ the reflexive closure of $\prec$.

3.3. The Enumeration Algorithm $\mathcal{M}$

The algorithm for the vertex v_0 (see Algorithm 1) is expressed in an event-driven description (see Tel [22] p. 553). The algorithm we describe here does not require FIFO communications, i.e., the emitted messages are not necessarily heard in the same order that they are emitted.

The action **I** can be executed by a process on wake-up only if it has not heard any message. In this case, it chooses the number 1, updates its mailbox and informs its neighbours.

The action **R** describes the instructions the vertex v_0 has to follow when it hears a message $< (n', n'_{old}, M') >$ from a neighbour. First, it updates its mailbox by adding M' to it. Then it modifies its number if there exists $(n(v_0), \ell, \mathcal{N}) \in M(v_0)$ such that $(\lambda(v_0), N(v_0)) \prec (\ell, \mathcal{N})$, i.e., if there exists another process in the network which has the same number with a greater local view. Then, it updates its local view according to the $update(n, n_{old})$ function described above. It adds its new state $(n(v_0), \lambda(v_0), N(v_0))$ to its mailbox. Finally, if its mailbox has been modified by the execution of all these instructions, it emits its number and its mailbox.

If the mailbox of v_0 is not modified by the execution of the action **R**, it means that the information v_0 has about its neighbour (i.e., its number) was correct, that all the elements of M' already belong to $M(v_0)$, and that for each $(n(v_0), \ell, \mathcal{N}) \in M(v_0)$, $(\ell, \mathcal{N}) \preceq (\lambda(v_0), N(v_0))$.

The action **S** is executed once the local boolean value $emit$ is set to $true$ by **I** or **R** actions. It means that the process needs to emit a message to all it neighbours.

3.4. Correctness of $\mathcal{M}$

Let **G** be a simple labelled graph. In the following, i is an integer denoting a computation step. Let $(\lambda(v), (n_i(v), N_i(v), M_i(v))$ be the label of the vertex v after the ith step of the computation of the algorithm $\mathcal{M}$ (Algorithm 1). We present some properties satisfied by each execution of the algorithm in the asynchronous broadcast model.

The following lemma, which can be proved easily by induction on the number of steps, recapitulates basic labelling properties.

Lemma 3.3. For each vertex v and each step i,

1. $n_i(v) \neq 0 \implies (n_i(v), \lambda(v), N_i(v)) \in M_i(v)$,
2. $\forall n' \in N_i(v)$ then $n' > 0$ and $\exists \ell' \in L, \exists N' \in \mathcal{P}_{\text{fin}}(\mathbb{N}^2)$ such that $(n', \ell', N') \in M_i(v)$.

The algorithm has some remarkable monotonicity properties that are described in the following lemma.

Lemma 3.4. For each step i and each vertex v, $M_i(v) \subseteq M_{i+1}(v)$, $n_i(v) \leq n_{i+1}(v)$, and $N_i(v) \preceq N_{i+1}(v)$. Moreover, if v applies the action **S** at step i and j with $i \neq j$, then $M_i(v) \neq M_j(v)$.

Proof:
The property is obviously true for the vertices that are not active at step i. It is easy to see that, for each vertex v, we always have $M_i(v) \subseteq M_{i+1}(v)$.

For each vertex v and each step i such that $n_i(v) \neq n_{i+1}(v)$, $n_{i+1}(v) = 1 + \max\{n_1; (n_1, \ell_1, N_1) \in M_i(v)\}$ and either $n_i(v) = 0 < n_{i+1}(v)$ or $(n_i(v), \lambda(v), N_i(v)) \in M_i(v)$ as shown in Lemma 3.3 and therefore $n_i(v) < n_{i+1}(v)$.

Algorithm 1: Algorithm $\mathcal{M}$ in the asynchronous broadcast model.

```
var:  $emit$ : bool init false;
      $n_{old}$ : int init $0$ ;
I : $\{n(v_0) = 0$ and no message has arrived at $v_0\}$
begin
    $M_{old} := \emptyset$;
    $n(v_0) := 1$ ;
    $M(v_0) := \{(n(v_0), \lambda(v_0), \emptyset)\}$;
    $emit := true$
end
S : $\{emit = true\}$
begin
    emit $< (n(v_0), n_{old}, M(v_0)) >$;
    $n_{old} := n(v_0)$;
    $emit := false$
end
R : {A message $< (n', n'_{old}, M') >$ has arrived at $v_0\}$
begin
    $M_{old} := M(v_0)$;
    $M(v_0) := M(v_0) \cup M'$;
    if $n(v_0) = 0$ or $\exists (n(v_0), \ell, \mathcal{N}) \in M(v_0)$ such that $(\lambda(v_0), N(v_0)) \prec (\ell, \mathcal{N})$ then
        $n(v_0) := 1 + \max\{n \mid \exists (n, \ell, \mathcal{N}) \in M(v_0)\}$;
    $N(v_0) := update(n', n'_{old})$;
    $M(v_0) := M(v_0) \cup \{(n(v_0), \lambda(v_0), N(v_0))\}$;
    if $\forall (n, p) \in N(v_0), p > 0$ and $M(v_0) \neq M_{old}$ then
        $emit := true$
end
```

When v hears a message in the following form: $mess =< (n', n', M') >$, $N_{i+1}(v) = update(n', n') = N_i(v)$. If $N_i(v) \neq N_{i+1}(v)$ then v heard a message $mess =< (n', n'_{old}, M') >$ with $n' > n'_{old}$ and thus $N_i(v) \prec N_{i+1}(v)$.

Moreover, the condition of **S** is satisfied when the value of $emit$ becomes true, i.e., when the mailbox $M(v)$ of v is modified. □

The local knowledge of a vertex v reflects to some extent some real properties of the current configuration. The two following lemmas enable us to prove that if a vertex v knows a number m (i.e., there exist ℓ, N such that $(m, \ell, N) \in M_i(v)$), then for each $m' \leq m$, there exists a vertex v' in the graph such that $n_i(v') = m'$. We first show that if v knows m there exists v' such that $n_i(v') = m$. we also show that if a vertex v knows an identity number m, then it knows all the numbers smaller than m.

Lemma 3.5. For each vertex $v \in V(G)$ and each step i, let $n_i(v) \neq 0$, given $(m', \ell', N') \in M_i(v)$, for every $1 \leq m \leq m'$, there exists a vertex $w \in V(G)$ such that $n_i(w) = m$ and $(m, \ell, N) \in M_i(v)$.

Proof:
By induction on step i, we show that for each vertex v with $n_i(v) \neq 0$, given $(m', \ell', N') \in M_i(v)$, for every $1 \leq m \leq m'$, there exists $(m, \ell, N) \in M_i(v)$. We state that it holds for all $i \geq 0$. If the rule I is applied by v, then, $M_i(v) = (1, \lambda(v_0), \emptyset)$ and trivially, the property holds.

If the rule R is applied by v, then, v heard a message $mess =< (n', n'_{old}, M') >$ from another vertex v'. Let j be the step in which v' emitted this message. We know that $M' = M_j(v')$. If v keeps its number at step $i+1$, then, $M_{i+1}(v) = M_i(v) \cup M_j(v')$ and the assertion is true by induction hypothesis. Besides, if v' modifies its number, then, $n_{i+1}(v) = 1 + max\{n \mid \exists (n, l, N) \in M_i(v) \cup M_j(v')\}$ and $M_{i+1}(v) = M_i(v) \cup M_j(v') \cup (n_{i+1}(v), \lambda(v), N_{i+1}(v))$. Consequently, the assertion is true.

Assume that the number m is known by v and let $U = \{(u, j) \in V(G) \times \mathbb{N} \mid j \leq i, n_j(u) = m\}$. Consider the set $U' = \{(u, j) \in U \mid \forall (u', j') \in U, N_{j'}(u') \prec N_j(u)$ or $N_{j'}(u') = N_j(u)$ and $j' \leq j\}$. It is easy to see that there exists i_0 such that for each $(u, j) \in U', j = i_0$. Since $(m, \ell, N) \in M_i(v)$, neither U nor U' are empty.

If $i_0 < i$, the number $n_{i_0}(u) = m$ of u was modified at step i_0+1 but by maximality of $(\lambda(u), N_{i_0}(u))$, the vertex u could not modify its number. Hence, $i_0 = i$ and there exists a vertex $w \in V(G)$ such that $n_i(w) = m$. □

From Lemma 3.5, we deduce that for each step, the identity numbers of all the vertices form either a set $[1, k]$ or a set $[0, k]$ with $k \leq V(G)$.

For each step i and each vertex v, if there exists $n' \in N_i(v)$, from Lemma 3.3, there exists v' such that $n_i(v') = n'$ and therefore $N(v)$ can only have a finite number of values and the same holds for $M(v)$. During the algorithm, the consecutive labelling of each vertex v form an increasing sequence, $(n_i(v), N_i(v), M_i(v))$, $i = 1, 2, \ldots$ and, each vertex can emit a message only if it modifies its mailbox. Since the number of possible accessible labels is finite (but dependent on the size of the graph), the algorithm always terminates.

Moreover, we make the assumption that every process knows the size of the network. Hence, once a process gets the number $|V(G)|$, from Lemma 3.5, it knows that all the vertices have different identity numbers that will not change anymore and it can locally detect the termination of the algorithm.

Since we have proven that $\mathcal{M}$ always terminates, we can give some properties about the final labelling:

Lemma 3.6. Any execution ρ of $\mathcal{M}$ on a connected labelled graph $\mathbf{G} = (G, \lambda)$ terminates and yields to a final labelling (λ, n_p, N_p, M_p) satisfying the following conditions:

1. there exists an integer $k \leq |V(G)|$ such that $\{n_p(v) \mid v \in V(G)\} = [1, k]$,

and for all vertices v, v':

2. $M_p(v) = M_p(v')$,
3. $(n_p(v), \lambda(v), N_p(v)) \in M_p(v')$,
4. $n_p(v) = n_p(v')$ implies that $\lambda(v) = \lambda(v')$ and $N_p(v) = N_p(v')$,

5. $(n, p) \in N_p(v)$ if and only if there exist $w_1, \ldots, w_p \in N_G(v)$ such that for each i, $n_p(w_i) = n$; in this case, there exists $(n_p(v), p') \in N_p(w_i)$ with $p' \geq 1$.

Proof:

1. By Lemma 3.5 applied to the final labelling.
2. Otherwise, there exist two neighbours v, v' such that $M(v) = M(v')$. However, since the configuration is final, both v and v' have sent their mailboxes to their neighbours and thus $M(v) = M(v')$.
3. A corollary of the previous point using Lemma 3.3.
4. A corollary of the previous property and since neither v nor v' need to change its number.
5. Since each neighbour of v that has the number n has sent a message with its number, and since all messages have been heard, we know that there exists $(n', p') \in N_p(v)$ with $p' > p$. Moreover, due to the design of the function replace, we know that $\sum_{(n,p)\in N_p(v), p>0} p$ is bounded by the degree of v. Consequently, the claim holds.

□

In the next proposition, we prove that there exists a digraph $\mathbf{D}$ associated to the final labelling of $\mathbf{G}$ such that $Dir(\mathbf{G})$ is a fibration of $\mathbf{D}$.

Proposition 3.7. Given a graph $\mathbf{G}$, we can associate to the final labelling of any execution ρ of the enumeration algorithm on $\mathbf{G}$, a digraph $\mathbf{D}$ such that $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$ and $V(D)$ is the set of numbers appearing on the vertices of $\mathbf{G}$ at the end of ρ.

Proof:
We use the notation of Lemma 3.6. Let $\mathbf{G} = (G, \lambda)$.

Consider the graph $\mathbf{D}$ defined as follows. Its set of vertices is $V(D) = \{m \in \mathbb{N} \mid \exists v \in V(G), n_\rho(v) = m\}$. For any $m, m' \in V(D)$, there are p arcs $a_{m',m,1}, \ldots, a_{m',m,p}$ from m' to m if there exists $v \in V(G)$ such that $n_\rho(v) = m'$ and $(m, p) \in N_\rho(v)$ with $p > 0$. From Lemma 3.6, this is independent of the choice of $v \in V(G)$. For every vertex $v, v' \in V(G)$, if $n_\rho(v) = n_\rho(v')$ then $\lambda(v) = \lambda(v')$ and we can define the labelling η of D: for every $v \in V(G)$, $\eta(n_\rho(v)) = \lambda(v)$.

Let us recall that $V(Dir(\mathbf{G})) = V(G)$ and for all edge $\{v, v'\} \in E(G)$, there exist two arcs $a_{v',v}, a_{v,v'}$ such that $s(a_{v,v'}) = t(a_{v',v}) = v$ and $t(a_{v,v'}) = s(a_{v',v}) = v'$. Moreover, for each $v \in V(G)$, the label of v in $Dir(\mathbf{G})$ is the same as in $\mathbf{G}$.

It remains to define the homomorphism φ from $Dir(\mathbf{G})$ to $\mathbf{D}$. For every vertex $v \in V(G)$, $\varphi(v) = n_\rho(v)$. For every vertex v such that $\varphi(v) = n$, and for each $(m, p) \in N_\rho(v)$ with $p > 0$, we know from Lemma 3.6 that there exist p arcs $a_1, \ldots, a_p \in A(Dir(\mathbf{G}))$ such that $t(a_i) = v$ and $n_\rho(s(a_i)) = m$. For each $1 \leq i \leq p$, $\varphi(a_i) = a_{m,n,i}$.

By definition, φ is a fibration and thus $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$. □

From Proposition 3.7, one can show that Algorithm $\mathcal{M}$ terminates on $\mathbf{G}$ and the final labelling verifies the following properties: $(Dir(G), \lambda)$ is fibred over $\mathbf{D}$. Thus if $Dir(\mathbf{G})$ is minimal then $\mathbf{D}$ is isomorphic to $Dir(\mathbf{G})$ and therefore the set of numbers of the vertices is exactly $1, \ldots, |V(\mathbf{G})|$: each vertex has a unique number. Moreover, we make the assumption that every process knows the size of the network. Hence, once a process has $|V(G)|$ different numbers in its mailbox, from Lemma 3.5, it knows that all the vertices have different identity numbers that will not change anymore.

Finally, we have proven the following theorem:

Theorem 3.8. For every graph $\mathbf{G}$, there exists a(n) naming/enumeration algorithm on $\mathbf{G}$ using asynchronous broadcast communications if and only if the digraph $Dir(\mathbf{G})$ is minimal.

3.5. Complexity Analysis

Complexity analysis of distributed algorithms constitutes a building block of many properties such as energy consumption when considering wireless sensor networks. In this part, we deal with the complexity of Algorithm 1. We are interested in the number of messages exchanged by the processes and their size. We also look at the memory needed by each vertex.

We consider that each vertex does not need to keep more than one element (n, ℓ, N) for each n in its mailbox. Indeed, if there are two elements $(n, \ell, N), (n, \ell', N') \in M(v)$, and if $(\ell, N) \prec (\ell', N')$, we can remove (ℓ, N) from the mailbox. Moreover, we assume that the initial labelling of G is such that each initial label ℓ can be encoded with $O(\log |V(G)|)$ bits.

Proposition 3.9. Let $\mathbf{G}$ be a labelled graph of size n with m edges and a maximum degree Δ. Any run of $\mathcal{M}$ yields $O(mn^2)$ emissions of messages of size $O(\Delta n \log n)$ bits. Moreover, it requires $O(\Delta n \log n)$ bits of memory at any vertex.

Proof:
Let $\mathbf{G}$ be a labelled graph of size n with m edges, maximal degree vertex Δ and diameter D. Consider a run ρ of the algorithm on $\mathbf{G}$. According to Lemma 3.5, we know that each vertex modifies its number at most n times.

For every vertex v, since numbers of v and of its neighbours only increase, $(n(v), N(v))$ can change $(d(v) + 1)n$ times. When v modifies its number or its local view, it yields at most the emission of $O(n)$ messages (because vertices that already have $(n(v), N(v))$ in their mailbox do not emit this message). Hence, any run of the algorithm needs $O(mn^2)$ messages. Since, each vertex only keeps useful informations in its mailbox, there exist at most n elements (n_0, ℓ, N) in $M(v)$ and each of these elements can be represented with $O(\Delta \log n)$ bits. Hence, one can represented the mailbox of each vertex with $O(\Delta n \log n)$ bits. Therefore, the size of each message is $O(\Delta n \log n)$ bits.

From these previous proofs, one knows that the mailbox of each vertex is encoded with $O(\Delta n \log n)$ bits. Moreover, for each vertex v, $n(v)$ can be represented with $\log n$ bits while $N(v)$ can be represented with $O(\Delta \log n)$ bits. Thus, the maximum local memory requirement at any vertex is $O(\Delta n \log n)$. $\square$

As a corollary of the complexity analysis, Theorem 3.8 is extended as follows:

Theorem 3.10. For every graph $\mathbf{G}$, there exists a polynomial complexity (memory, messages and size of messages) naming/enumeration algorithm on $\mathbf{G}$ using asynchronous broadcast communications if and only if the digraph $Dir(\mathbf{G})$ is minimal.

Algorithms of Yamashita and Kameda and of Boldi *et al.* presented in Section 1.4 yields $O(n^2)$ emissions of messages of size $2^{O(n)}$ bits. Moreover, each process requires $2^{O(n)}$ memory bits. Thus, considering different aspects of the complexity, $\mathcal{M}$ fits particularly well to multi-hop broadcast networks composed with low-capabilities processes (e.g., wireless sensors).

4. A Leader Election Algorithm for Broadcast Networks

As stated in the introduction, if we can solve the enumeration problem on a graph $\mathbf{G}$ then we can solve the election problem on this graph by declaring the vertex with the identity number $|V(G)|$ as elected. Nonetheless, in our model, the enumeration and the election problems are not equivalent. Consider the graph $\mathbf{G}$ and the digraph $Dir(\mathbf{G})$ of Figure 1. Since $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$, from Theorem 3.8, the enumeration problem cannot be solved on $\mathbf{G}$. Nonetheless, if every vertex initially knows $\mathbf{G}$, consider a leader election algorithm defined as follows: each vertex emits a message and, once a vertex receives four messages, it can declare itself as elected. Since the vertex labelled 3 is the unique vertex of degree greater or equal than 4 in $\mathbf{G}$, the vertex 3 will be elected.

In this section, we also present an impossibility result which states that there exists no leader election algorithm for a graph $\mathbf{G}$ if $Dir(\mathbf{G})$ is not nt-minimal. This condition is sufficient and we give an extension of $\mathcal{M}$ (Algorithm 2) which solves the election problem.

4.1. Impossibility Result

Given a network represented by a simple graph $\mathbf{G}$, we present a necessary condition based on nt-fibrations that must be satisfied by $\mathbf{G}$ to admit a leader election algorithm.

Proposition 4.1. Let $\mathbf{G}$ be a labelled graph such $Dir(\mathbf{G})$ is not nt-minimal, there is no leader election algorithm for $\mathbf{G}$ in the asynchronous broadcast model.

Proof:
Consider a simple graph $\mathbf{G} = (G, \lambda)$ and a strongly connected digraph $\mathbf{D} = (D, \eta)$ such that $Dir(\mathbf{G})$ is nt-fibred over $\mathbf{D}$ via a fibration φ. Given an algorithm $\mathcal{A}$ using asynchronous broadcast communications, consider an execution of $\mathcal{A}$ on $\mathbf{D}$ as described in Lemma 2.11. Note that if there exists an infinite execution of $\mathcal{A}$ on $\mathbf{D}$, then, following Lemma 2.11, there exists an infinite execution of $\mathcal{A}$ on $\mathbf{G}$. Finally, $\mathcal{A}$ is not a leader election algorithm for $\mathbf{G}$.

Suppose that there exists a finite and maximal execution of $\mathcal{A}$ on $\mathbf{D}$ which yields a digraph $\mathbf{D}'$. In the final configuration every message has arrived and no process has to emit a message. Thus, each vertex has its final label. Following Lemma 2.11, there exists a lifted execution of $\mathcal{A}$ on $Dir(\mathbf{G})$ that yields a configuration $\mathbf{G}'$ such that $\mathbf{G}'$ is fibred over $\mathbf{D}'$ via φ. Since $\mathbf{G}'$ is nt-fibred over $\mathbf{D}'$, it implies that for every vertex $v \in V(\mathbf{G})$, there exist at least two vertices in $\varphi^{-1}(\varphi(v))$ that have the same label in $\mathbf{G}'$. Hence, there exists no vertex $v \in V(\mathbf{G})$ that has a unique label. The algorithm $\mathcal{A}$ is not a leader election algorithm for $\mathbf{G}$. □

4.2. Initial Knowledge

We here underline the importance of the initial knowledge. In the previous algorithm $\mathcal{M}$, every process only knows the size of the network. Using this initial knowledge, we ensure that at the end of the execution, each process locally knows that each vertex has obtained a unique identity even though some messages are arbitrarily delayed. Boldi *et al.* [5] and Yamashita and Kameda [24] also show that knowing the size of the graph allows to solve election problem whenever it is possible. However, in their models, each vertex initially knows its degree (or can compute it easily) and this initial knowledge is actually used in their views construction algorithm.

In our model, vertices do not initially know their degree and in this case, the initial knowledge of the size of the graph is not sufficient to solve the election problem on graphs where it can be solved. For instance, assume that there exists a leader election algorithm for the three graphs $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$ of Figure 2. In $\mathbf{G}_1$ (resp. $\mathbf{G}_2$, $\mathbf{G}_3$), there exists a unique vertex of degree 4 (resp. 5, 5). Hence, similarly to the graph of Figure 1, one can elect in these three graphs when we assume that each process initially knows the graph. Consider the digraph $\mathbf{B}$ such that $Dir(\mathbf{G}_1)$ is ι-fibred over $\mathbf{B}$ via a fibration φ. When executed on $\mathbf{B}$, a leader election algorithm for $\mathbf{G}_1$ has to elect a process such that its fibre is trivial. Thus, there exist two vertices $a, b \in \mathbf{B}$ such that $|\varphi^{-1}(a)| = |\varphi^{-1}(b)| = 1$ and which can be declared as elected. Assume that several messages are arbitrary delayed, i.e., several communication links are not yet established. One can find two graphs $\mathbf{G}_2$ and $\mathbf{G}_3$ and two digraphs $\mathbf{D}_2$ and $\mathbf{D}_3$ such that $\mathbf{D}_2 \subseteq Dir(\mathbf{G}_2)$ and $\mathbf{D}_3 \subseteq Dir(\mathbf{G}_3)$ and such that $\mathbf{D}_2$ and $\mathbf{D}_3$ are also ι-fibred over $\mathbf{B}$.

From Lemma 2.11, if there exists a finite and maximal execution of an algorithm that elects a leader in $\mathbf{B}$ then there exists a finite and maximal execution on $Dir(\mathbf{G}_1)$, $\mathbf{D}_2$ and $\mathbf{D}_3$ that also elects a leader. Hence, if the vertex b is declared as elected in $\mathbf{B}$, then there exists an execution on $Dir(\mathbf{G}_2)$ where messages sent along arcs in $Dir(\mathbf{G}_2) \setminus \mathbf{D}_2$ are delayed for an arbitrary long time. At some point in this execution, two vertices have the final label *elected*. Similarly, if the vertex a is declared as elected in $\mathbf{B}$, then there exists a particular execution on $Dir(\mathbf{G}_3)$ such that two vertices are marked as elected. Therefore, we cannot find a universal leader election algorithm for all graphs of order 8 where election problem can be solved. In the following, we provide a leader election algorithm $\mathcal{M}_e$ which assumes that each process knows a map of the network.

4.3. Informal Description of the Leader Election Algorithm

We present how to extend $\mathcal{M}$ to solve the leader election problem on digraphs that are $\pi\iota$-minimal.

Consider a graph $\mathbf{G}$ such that $Dir(\mathbf{G})$ is ι-fibred over a digraph $\mathbf{D}$. Our aim is to provide an extension of our previous algorithm by using the termination detection algorithm of [20]. The idea is to execute this algorithm and to reconstruct a graph from the contents of the vertices mailboxes (as it is done in Proposition 3.7) and check if all processes are involved in the execution, i.e., if there is no isolated process.

4.3.1. The SSP Algorithm

Initially in [20], this algorithm was devised to detect the termination of another distributed algorithm. As stated in Section 3.4, each process is able to determine its termination condition. The SSP algorithm detects an instant in which the entire computation is achieved.

Let G be a graph, to each process v is associated a predicate $P(v)$ and an integer $a(v)$, its confidence level. Initially, $P(v)$ is false and $a(v)$ is equal to -1. If a vertex v has finished its computation of the initial algorithm, then it changes its value $P(v)$ to true. Each time a vertex changes the value of $P(v)$ or $a(v)$ then it informs its neighbours.

The modification of the value of $a(v_0)$ only depends on the value of $P(v_0)$ and the informations v_0 has about the values $\{a(v_1), \ldots, a(v_d)\}$ of its neighbours:

- if $P(v_0) = false$ then $a(v_0) = -1$,
- if $P(v_0) = true$ then $a(v_0) = 1 + min\{a(v_k) \mid k \in [0; k]\}$.

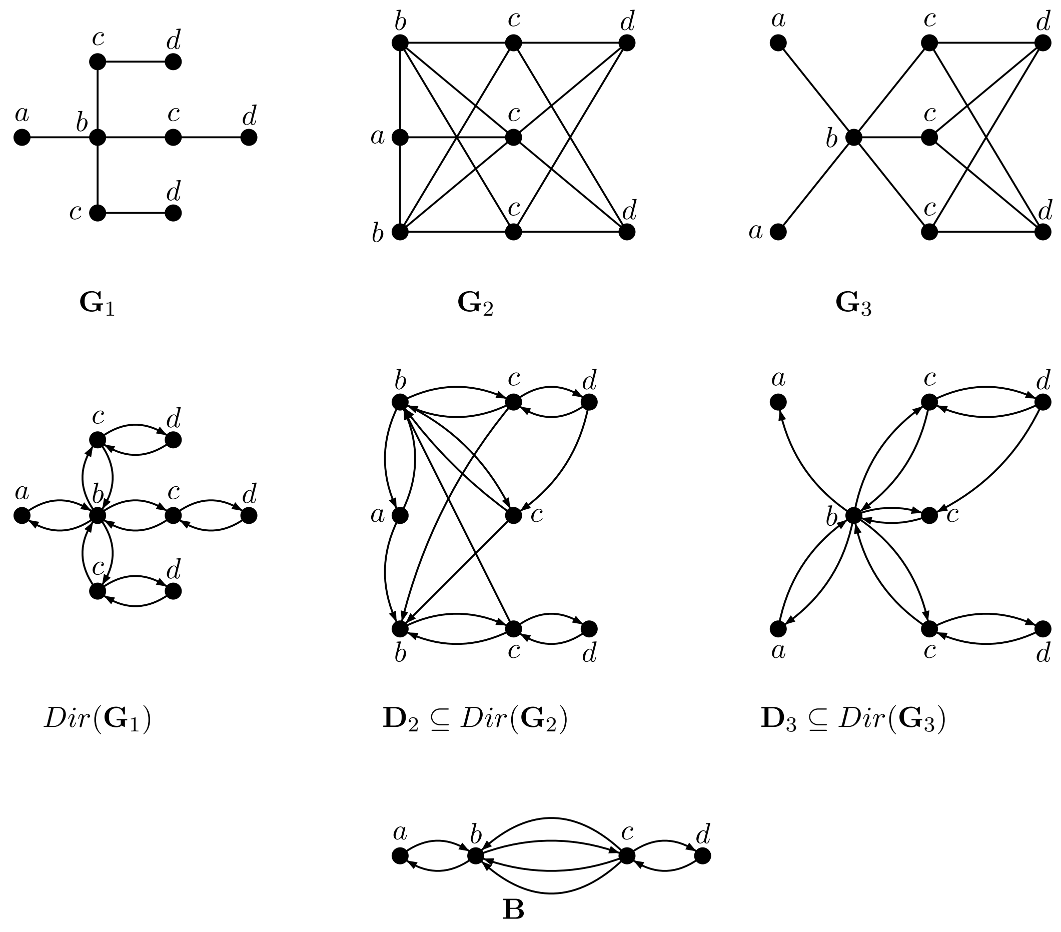
$\mathbf{G}_1$
$\mathbf{G}_2$
$\mathbf{G}_3$
$Dir(\mathbf{G}_1)$
$\mathbf{D}_2 \subseteq Dir(\mathbf{G}_2)$
$\mathbf{D}_3 \subseteq Dir(\mathbf{G}_3)$
$\mathbf{B}$
a
b
c
d

Figure 2. The labelled digraph $Dir(\mathbf{G}_1)$ is fibred over the digraph $\mathbf{B}$. This fibration is a ι-fibration and $Dir(\mathbf{G}_1)$ is $n\iota$-minimal; the subdigraphs $\mathbf{D}_2$ of $Dir(\mathbf{G}_2)$ and $\mathbf{D}_3$ of $Dir(\mathbf{G}_3)$ are also ι-fibred over the minimal base $\mathbf{B}$. From Lemma 2.11, an execution of a leader election algorithm on $\mathbf{B}$ can be lifted to an execution on $Dir(\mathbf{G}_1)$ and an execution on $\mathbf{D}_2$ and $\mathbf{D}_3$. Thus, the vertex a can be declared as elected in $\mathbf{B}$, $\mathbf{G}_1$ and $\mathbf{G}_2$ and the vertex b can be declared as elected in $\mathbf{B}$, $\mathbf{G}_1$ and $\mathbf{G}_3$. If the algorithm chooses a (resp. b), then two vertices in $\mathbf{G}_2$ (resp. $\mathbf{G}_3$) are declared as elected: that is not possible.

We will adapt this algorithm using the ideas of the algorithm GSSP [12]. For every vertex v, the value of $P(v)$, instead of being boolean, will be a graph reconstructed from the contents of the mailbox of v. An important property of the function P is that it is constant between two moments where it has the same value.

In our models, a vertex cannot distinguish its neighbours: therefore we will use the numbers that appear in the local view. A vertex v will increase its confidence level $a(v)$ only if when $|N(v)| = k$, then v has heard messages from k different processes v' such that $M(v') = M(v)$ and $a(v') \geq a(v)$.

In our algorithm, each vertex permanently tries to reconstruct a digraph $\mathbf{D}(M)$ from its mailbox. This digraph is constructed as in Proposition 3.7. Given a mailbox M, we say that an element $(n, \ell, N) \in M$ is *maximal* if for all $(n, \ell', N') \in M$, $(\ell', N') \preceq (\ell, N)$; we denote by $\max(M)$ the set of maximal elements of M; note that for each n, there is at most one element $(n, \ell, N) \in \max(M)$. If there exists

$(n, \ell, N) \in \max(M)$ such that there is $(m, p) \in N$ with $p < 0$, or if there is no $(m, \ell', N') \in \max(M)$, then $\mathbf{D}(M)$ is undefined. Otherwise, the digraph $\mathbf{D}(M)$ is defined as follows: $V(D(M)) = \{n \mid \exists(n, \ell, N) \in \max(M)\}$, and for each $(n, \ell, N) \in \max(M)$, $\lambda(n) = \ell$, and for each $(m, p) \in N$, there are exactly p arcs from m to n in $\mathbf{D}(M)$.

4.3.2. Labels

As in the enumeration algorithm, we start with a labelled graph $\mathbf{G} = (G, \lambda)$. During the computation, vertices v will get new labels of the form $(\lambda(v), n(v), N(v), M(v), a(v), A(v))$. Thus, we add to the label of each vertex two items:

- $a(v) \in \mathbb{N}$ is the confidence level of the vertex v,
- $A(v) \in \mathcal{P}_{\text{fin}}(\mathbb{N} \times \mathbb{Z} \times \mathbb{Z})$ is a set maintained by each vertex v. It contains the confidence level of its neighbours in the form (n, p, a) where p is the number of the neighbours of v with n as identity number and a as confidence level.

For sake of simplicity, we define a function $confidence(n, a)$ to update the set $A(v_0)$ of a process v_0 as follows. First, if $a \geq 0$, we let $a_{old} = a - 1$ and we apply the following rules:

- if there exists $(n, 1, a_{old}) \in A(v_0)$ then $A(v_0) := A(v_0) \setminus \{(n, 1, a_{old})\}$,
- if there exists $(n, p, a_{old}) \in A(v_0)$ with $p \neq 1$ then $A(v_0) := A(v_0) \setminus \{(n, p, a_{old})\} \cup \{(n, p - 1, a_{old})\}$,
- otherwise, $A(v_0) := A(v_0) \cup \{(n, -1, a_{old})\}$.

Then, symmetrically, we do the following operations:

- if there exists $(n, -1, a) \in A(v_0)$ then $A(v_0) := A(v_0) \setminus \{(n, -1, a)\}$,
- if there exists $(n, p, a) \in A(v_0)$ with $p \neq -1$ then $A(v_0) := A(v_0) \setminus \{(n, p, a)\} \cup \{(n, p + 1, a)\}$,
- otherwise, $A(v_0) := A(v_0) \cup \{(n, 1, a)\}$.

Note that in Algorithm 2, the digraph $\mathbf{B_G}$ is the minimal base of the initial digraph $Dir(\mathbf{G})$ on which the algorithm is performed.

4.3.3. Messages

A message emitted by a process u and heard by the process v has the following form $< (m, n_{old}, M, a) >$ where m, n_{old} and M are identical to values of messages exchanged in $\mathcal{M}$. We add the item a which is the value of the confidence level $a(u)$ of a.

4.4. The Leader Election Algorithm $\mathcal{M}_e$

The algorithm for the vertex v_0 is described in Algorithm 2. The core of the action $\mathbf{I}$ remains unchanged compared to Algorithm $\mathcal{M}$ except that the vertex v_0 has to initialize its confidence level $a(v_0)$ to -1.

The action $\mathbf{R}$ contains the instructions the vertex v_0 has to follow when it heard a message $< (n', n'_{old}, M', a') >$ from a neighbour. Initially, it behaves as in Algorithm $\mathcal{M}$. If its mailbox has been modified, it has to reset its confidence level $a(v_0)$ and the confidence it has collected from its neighbours in $A(v_0)$. Conversely, if its mailbox remains unchanged, it updates $A(v_0)$ with the received value a'. In order to update its own confidence level, the vertex v_0 verifies if every collected confidence levels in $A(v_0)$ are greater than $a(v_0)$ and if the graph reconstructed from its mailbox $M(v_0)$ is fibred over $\mathbf{B_G}$. It means that v_0 has the same mailbox of its neighbours. Following these instructions, if its mailbox or its confidence level has been modified by the execution, it emits its number with its new mailbox and confidence level. Finally, an execution is terminated when its confidence level is greater than the size of the graph $|V(G)|$. It means that all processes have reconstructed the same graph from their mailbox which is fibred over $\mathbf{B_G}$. Thus, the vertex whose its fibre is trivial is declared as elected. If there exist more than one vertex satisfying this condition, the vertex with the lowest number is chosen.

4.5. Correctness of $\mathcal{M}_e$

Let $\mathbf{G}$ be a simple labelled and connected graph. In the following, i is an integer denoting a computation step. Let $(\lambda(v), n_i(v), N_i(v), M_i(v), a_i(v), A_i(v))$ be the label of the vertex v after the ith step of the computation of the algorithm $\mathcal{M}_e$. We present some properties satisfied by each execution of the algorithm in the asynchronous broadcast model.

We can easily state by induction that if the mailbox of a vertex v is the same between two steps, the confidence level of v increases.

Lemma 4.2. For each step i and each vertex v, if $M_i(v) = M_{i+1}(v)$ then $a_{i+1}(v) \geq a_i(v)$. Moreover, if v applies the action $\mathbf{S}$ at steps i and j, then $M_i(v) \neq M_j(v)$ or $a_i(v) \neq a_j(v)$.

In the following lemma, we show that when a process emits a message, then $\forall (n,p,a) \in A(v)$, $a \geq a(v) - 1$.

Lemma 4.3. For each step i and each vertex v, either $\forall (n,p,a) \in A_i(v)$, $a \geq a_i(v) - 1$, or there exists $(n,p,a) \in A_i(v)$ such that $p < 0$ and $\forall (n,p',a') \in A_i(v)$, $a' \geq a$.

Proof:
We prove the lemma by induction on i. Initially, $A(v) = \emptyset$ and the property obviously holds. Suppose that the property holds for all vertices at step i and consider a vertex v that hears a message $< n', n'_{old}, M', a' >$ at step $i+1$. If $n'_{old} \neq n'$, or if $M' \neq M_i(v)$, then $a_{i+1}(v) = -1$ and for all $(n,p,a) \in A_{i+1}(v)$, $a = -1$. Note that if $M_{i+1}(v) = M_i(v)$ and $a_{i+1}(v) \neq a_i(v)$, then $a_{i+1}(v) = 1 + \min\{a \mid \exists (n,p,a) \in A_{i+1}(v)\}$ and the property holds.

Suppose that at step i, $\forall (n,p,a) \in A_i(v)$, $a \geq a_i(v)$. If $a' \geq a_i(v)$, then $\forall (n,p,a) \in A_{i+1}(v)$, $a \geq a_{i+1}(v) - 1$. If $a' \leq a_i(v) - 1$, then there exists $(n', -1, a'-1) \in A_{i+1}(v)$ and $\forall (n', p'', a'') \in A_i(v)$, $a'' \geq a' - 1$.

Suppose now that at step i, there exists $(n,p,a) \in A_i(v)$ such that $p < 0$ and $\forall (n, p'', a'') \in A_i(v)$, $a'' \geq a$. If $n' \neq n$, then the property still holds. Otherwise, if $a' \leq a$, then $(n', -1, a'-1) \in A_{i+1}(v)$ and $\forall (n, p'', a'') \in A_{i+1}(v)$, $a'' \geq a' - 1$; If $a' = a + 1$, then $(n', p-1, a) \in A_{i+1}(v)$ and $\forall (n, p'', a'') \in A_{i+1}(v)$, $a'' \geq a$; If $a' > a + 1$, then $(n', p, a) \in A_{i+1}(v)$ and $\forall (n, p'', a'') \in A_{i+1}(v)$, $a'' \geq a$. □

Consider a vertex $v \in V(\mathbf{G})$ and a step i, for any given $a \geq 0$, for every $(n,p) \in N_i(v)$, let $\mathcal{X}_i(n,a,v) = \{p' \mid \exists (n,p',a') \in A_i(v) \text{ such that } a' \geq a\}$ and $x_i(n,a,v) = \sum_{p \in \mathcal{X}_i(n,a,v)} p$.

Algorithm 2: Algorithm $\mathcal{M}_e$ in the asynchronous broadcast model.

```
var:   $emit$ : bool init false;
I : $\{n(v_0) = 0$ and no message has arrived at $v_0\}$
begin
    $n(v_0) := 1$; $a(v_0) := -1$;
    $M_{old} := \emptyset$; $a_{old} := -1$; $n_{old} := 0$;
    $M(v_0) := \{(n(v_0), \lambda(v_0), \emptyset)\}$;
    $emit := true$
end

S : $\{emit = true\}$
begin
    if $\forall (n,p) \in N(v_0), p > 0$ and $\forall (n,p,a) \in A(v_0), p > 0$ then
        if $a(v_0) = -1$ then
            emit $< (n(v_0), n_{old}, M(v_0), a(v_0)) >$;
        else while $a_{old} < a(v_0)$ do
            $a_{old} := a_{old} + 1$ ;
            emit $< (n(v_0), n_{old}, M(v_0), a_{old}) >$;
        $emit := false$ ; $n_{old} := n(v_0)$ ; $a_{old} := a(v_0)$;
end

R : {A message $< (n', n'_{old}, M', a') >$ has arrived at $v_0\}$
begin
    $M_{old} := M(v_0)$;
    $M(v_0) := M(v_0) \cup M'$;
    if $n(v_0) = 0$ or $\exists (n(v_0), \ell, \mathcal{N}) \in M(v_0)$ such that $(\lambda(v_0), N(v_0)) \prec (\ell, \mathcal{N})$ then
        $n(v_0) := 1 + \max\{n \mid \exists (n, \ell, \mathcal{N}) \in M(v_0)\}$;
    $N(v_0) := update(n', n'_{old})$;
    $M(v_0) := M(v_0) \cup \{(n(v_0), \lambda(v_0), N(v_0))\}$;
    if $M(v_0) \neq M_{old}$ then
        $a(v_0) := -1$; $a_{old} := -1$;
        $A(v_0) := \{(n, p, -1) \mid (n,p) \in N(v_0)\}$;
    if $M(v_0) = M'$ and $a' \geq 0$ then
        $A(v_0) := confidence(n', a')$;
    if $\forall (n,p,a) \in A(v_0), a(v_0) \leq a$ then
        construct $\mathbf{D}(M(v_0))$ from $M(v_0)$;
        if $\mathbf{D}(M(v_0))$ is fibred over $\mathbf{B_G}$ then
            $a(v_0) := 1 + \min\{a \mid \exists (n,p,a) \in A(v_0)\}$;
    if $a(v_0) \neq a_{old}$ or $M(v_0) \neq M_{old}$ then
        $emit := true$;
    if $a_i(v) > |V(G)|$ then
        compute $C_{\mathbf{G},\mathbf{D}(M(v_0))}$;
        if $n(v_0) = \min\{n \mid n \in C_{\mathbf{G},\mathbf{D}(M(v_0))}\}$ then $status :=$ elected;
        else $status :=$ non-elected;
end
```

Lemma 4.4. Consider a step i. For every vertex $v \in V(\mathbf{G})$ and any given $a \geq 0$, if $k = x_i(n, a, v) > 0$, there exist k neighbouring vertices $w_1, \ldots, w_k \in Dir(\mathbf{G})$ such that for every $0 < l \leq k$, v has heard a message $< (n, n', M, a) >$ from w_l before step i.

Proof:
Assume that $a = a_{max} = max\{a' \mid (n, p, a') \in A_i(v)\}$. Thus, $\mathcal{X}_i(n, a_{max}, v) = \{p' \mid \exists (n, p', a') \in A_i(v)$ such that $a' = a_{max}\}$ and $x_i(n, a_{max}, v) = p'$. This means that the process v has heard p' messages in the form $< (n, n_{old}, M, a) >$ before step i. By Lemmas 3.4 and 4.2, we deduce that the assertion is satisfied.

Consider $a < a_{max}$. Suppose that the assertion holds for $x_i(n, a+1, v)$. Hence, v has heard at least $x_i(n, a+1, v)$ messages **mess**$=< (n, n_{old}, M, a+1) >$. Thus, from Lemma 4.3, for each message **mess** heard by v, the $confidence(n, a+1)$ function is called and an element (n, a) is removed from $A_i(v)$. This means that if $(n, p', a) \in A_i(v)$, the process v has heard $p' + x_i(n, a+1, v)$ messages $< (n, n_{old}, M, a) >$ before step i. By Lemmas 3.4 and 4.2, each of these messages has been emitted by a different neighbour of v. Therefore, the property is verified. □

Consider a step i_0 and a vertex v_0 such that $a_{i_0}(v_0) \geq 0$. We denote $M = M_{i_0}(v_0)$. For every vertex $v \in V(\mathbf{G})$, we define $i(v, M, i_0)$ (or $i(v)$ when it is clear from the context) as follows. If there is a step i such that v emits a message $< n_i(v), n_{old}, M_i(v), a_i(v) >$ with $M_i(v) = M$, then $i(v)$ is the last step where v emits a message of this form; otherwise $i(v) = \infty$. We define a digraph $\mathbf{H}(M, i_0)$ as follows. For every vertex $v \in V(Dir(\mathbf{G}))$, v belongs to $V(\mathbf{H}(M, i_0))$ if $i(v) < \infty$. For each vertex $v \in V(\mathbf{H}(M, i_0))$, for every $(n, p) \in N_{i_0}(v)$, let $k = x_{i_0}(n, a_{i(v)}(v) - 1, v)$. From Lemma 4.4, there exist k neighbouring vertices $w_1, \ldots, w_k$ of v such that for every $0 < l \leq k$, $w_l \in V(\mathbf{H}(M, i_0))$ and $n_{i_0}(w_l) = n$ and v has heard a message $< (n, n_{old}, M, a_{i(v)}(v) - 1) >$ from w_l before step $i(v)$. Each corresponding arc from w_l to v belongs to $A(\mathbf{H}(M, i_0))$. In the following, we prove that while $\mathbf{H}(M, i_0) \neq Dir(\mathbf{G})$, then the execution of the algorithm is not terminated.

For every vertex v, since $a(v)$ and the number of given identities are bounded by $|V(G)|$, we know that any execution of $\mathcal{M}_e$ terminates. In the next lemma, we show that the confidence level of a vertex allows to know how far from v the vertices have the same mailbox as v.

Lemma 4.5. Consider a step i_0 and a mailbox M. For all vertices $v, w \in V(\mathbf{H}(M, i_0))$, if $dist_{\mathbf{H}(M,i_0)}$ $(w, v) \leq a_{i(v)}(v)$, then $a_{i(w)}(w) \geq a_{i(v)}(v) - dist_{\mathbf{H}(M,i_0)}(w, v)$.

Proof:
Let $\mathbf{H} = \mathbf{H}(M, i_0)$. This lemma can be proved by induction on the distance d between w and v in $\mathbf{H}$. Assume that $d = 1$. Hence, $a_{i(v)}(v) \geq dist_{\mathbf{H}}(w, v) \geq 1$ and $w \in N_{\mathbf{H}}(v)$. Since $a_{i(v)}(v) \geq 1$, we know that for all $(m, p, a) \in A_{i(v)}(v)$, $a \geq a_{i(v)}(v) - 1$. Thus, from the definition of $H(M, i)$ and Lemma 4.3, for every vertex $w \in N_{H(M,i)}(v)$, w has sent a message $< (n(w), n_{old}(w), M, a_{i(v)}(v) - 1) >$. Consequently, for each $w \in N_{H(M,i)}$, there exists a step $j < i(v) \leq i_0$ such that $M_j(v) = M$ and $a_j(w) \geq a_{i(v)}(v) - 1$, and thus $a_{i(w)}(w) \geq a_{i(v)}(v) - 1$.

We assume that it holds for every vertex v, w such that $dist_{\mathbf{H}}(w, v) \leq d$. Consider two vertices v, w such that $a_{i(v)}(v) \geq d+1$ and $dist_{\mathbf{H}}(w, v) = d+1$. Consider a vertex $u \in \mathbf{H}$ such that $(w, u) \in A(H)$ and $dist_{\mathbf{H}}(u, v) = d$. By induction hypothesis, $a_{i(u)}(u) \geq a_{i(v)}(v) - d$ and $a_{i(w)}(w) \geq a_{i(u)}(u) - 1$. Consequently, $a_{i(w)}(w) \geq a_{i(v)}(v) - (d+1)$. □

Let us recall that $\mathbf{B_G}$ is the digraph such that $Dir(\mathbf{G})$ is ι-fibred over $\mathbf{B_G}$ via a fibration relation φ and $\mathbf{B_G}$ is the minimal base of $Dir(\mathbf{G})$. When one considers an execution of $\mathcal{M}_e$ in which some messages are delayed, every process involved in the computation belongs to a subdigraph $\mathbf{H}$ of $Dir(\mathbf{G})$. In the following lemma, we show that when $\mathbf{H}$ is fibred over $\mathbf{B_G}$, the view of each vertex $v \in V(H)$ is isomorphic to the view of $v \in V(G)$.

Lemma 4.6. Let $\mathbf{H}$ be a subdigraph of $Dir(\mathbf{G})$ and the digraph $\mathbf{B_G}$ such that $Dir(\mathbf{G})$ (resp. $\mathbf{H}$) is fibred over $\mathbf{B_G}$ via a fibration relation $\varphi_\mathbf{G}$ (resp. $\varphi_\mathbf{H}$). If x_0 is the vertex with the maximal view in $\mathbf{B_G}$, then $\varphi_\mathbf{H}(v) = x_0 \implies \varphi_\mathbf{G}(v) = x_0$. Moreover, for every vertex $v \in \mathbf{H}$, $T_\mathbf{G}(v) \approx T_\mathbf{H}(v)$ and thus $\mathbf{H} \approx \mathbf{G}$.

Proof:
Since $\mathbf{H}$ is a subdigraph of $\mathbf{G}$, from Remark 2.9, for each v, $T_\mathbf{H}(v) \preceq T_\mathbf{G}(v)$. Since $\mathbf{H}$ is fibred over $\mathbf{B_G}$ via φ, for every w_0 in $\mathbf{B_G}$ that has a maximal view, for every $v_0 \in \varphi^{-1}(w_0)$, $T_\mathbf{H}(v_0)$ is maximal in $\mathbf{G}$ and thus $T_\mathbf{H}(v_0) = T_\mathbf{G}(v_0)$.

We now prove that for every vertex v in $V(\mathbf{G})$, $T_\mathbf{H}(v) = T_\mathbf{G}(v)$. Let X_0 be the set of vertices that have a maximal view. Let v_0 be the closest vertex from v in $\mathbf{G}$ such that $T_\mathbf{G}(v_0)$ is maximal, and let $dist_\mathbf{G}(v, X_0)$ be the distance from v to v_0 in $\mathbf{G}$. We prove the result by induction on $dist_\mathbf{G}(v, X_0)$. If $v \in X_0$, then we already know the result holds. Otherwise, there exists a neighbour u of v such that $dist_\mathbf{G}(u, X_0) = dist_\mathbf{G}(v, X_0) - 1$. By induction, we know that $T_\mathbf{G}(u) \approx T_\mathbf{H}(u)$, and thus u has the same degree in $\mathbf{G}$ and in $\mathbf{H}$. Moreover, the multiset of the views of the neighbours of u should be the same in $\mathbf{H}$ and $\mathbf{G}$. Consequently, if $T_\mathbf{H}(v) \prec T_\mathbf{G}(v)$, there exists another neighbour v' of v such that $T_\mathbf{G}(v) \prec T_\mathbf{H}(v)$, which is impossible. Thus, for any $v \in V(H)$, $T_\mathbf{G}(v) \approx T_\mathbf{H}(v)$ and $N_\mathbf{G}(v) = N_\mathbf{H}(v)$. Since $\mathbf{G}$ is connected, $V(G) = V(H)$ and $Dir(\mathbf{G}) \approx \mathbf{H}$. □

From Proposition 3.7, once the enumeration algorithm is terminated on $\mathbf{H}(M, i_0)$, every vertex v has the same mailbox $M = M(v)$ and is able to construct a labelled digraph $\mathbf{D}(M(v))$. We have to show that if $\mathbf{D}(M(v))$ is fibred over $\mathbf{B_G}$, then $\mathbf{H}(M, i_0) = Dir(\mathbf{G})$.

We now prove in the following lemma that once a vertex gets a confidence level greater than the size of the graph, all vertices of the graph have the same mailbox and have a confidence level greater than 0.

Lemma 4.7. If there exist a step i_0 and a vertex v such that $a_{i_0}(v) > |V(G)|$, then there exists a subdigraph $\mathbf{H}'$ of $\mathbf{H}(M_{i_0}(v), i_0)$ such that $\mathbf{H}'$ is fibred over $\mathbf{D}(M_{i_0}(v))$.

Proof:
Let $M = M_{i_0}(v)$ and consider the graph $\mathbf{H}(M, i_0)$ defined above and let V' be the set of vertices $w \in V(H(M, i_0))$ such that there exists a path from w to v in $\mathbf{H}(M, i_0)$. Let $\mathbf{H}'$ be the subgraph of $\mathbf{H}(M, i_0)$ induced by V'. From Lemma 4.5, for each $w \in V(H')$, $M_{i(w)}(w) = M$ and $a_{i(w)}(w) \geq 1$. Since $a_{i(w)}(w) \geq 1$, there does not exist $(n_{i(w)}(w), \ell', N') \in M$ such that $(\lambda(w), N_{i(w)}(w)) \prec (\ell', N')$. Consequently, for all $w, w' \in V(H')$, if $n_{i(w)}(w) = n_{i(w')}(w')$, then $\lambda(w) = \lambda(w')$ and $N_{i(w)}(w) = N_{i(w')}(w')$.

Note that since $a_{i(w)}(w) \geq 1$, for every $(n, p, a) \in A_{i(w)}(w)$, $a \geq 0$. Consequently, for every $(n, p) \in N_{i(w)}(w)$, $x_{i(w)}(n, a_{i(w)}(w) - 1, w) = p$. Consequently, in $\mathbf{D}(M)$, for every $(n, p) \in N_{i(w)}(w)$, there are p arcs from the vertex n to the vertex $n_{i(w)}(w)$.

We define a homomorphism φ from $\mathbf{H}'$ to $\mathbf{D}(M)$ as follows. For each vertex $w \in V(H')$, let $\varphi(w) = n_{i(w)}(w)$. Considering a vertex $w \in V(H')$, we define the image by φ of all its incoming arcs as follows. By construction of $\mathbf{H}(M, i_0)$, for each $(n,p) \in N_{i(w)}(w)$, we know that there exist exactly $a_1, \ldots, a_p \in A(H(M, i_0))$ such that for each $l \in [1,p]$, $t(a_l) = w$ and $n_{i(s(a_l))}(s(a_l)) = n$. Thus, we let $\varphi(a_l) = a_{n,n_{i(w)}(w),l}$. By construction, $\mathbf{H}'$ is fibred over $\mathbf{D}(M)$ via φ. □

Thus, if there exists a vertex v such that the digraph $\mathbf{D}(M(v))$ reconstructed from its mailbox $M(v)$ is not fibred over the minimal base $\mathbf{B_G}$ of $Dir(\mathbf{G})$, the algorithm is not terminated.

In the following lemma, we show that, at the end of any execution of $\mathcal{M}_e$ on a nt-minimal graph, only one vertex is declared as *elected*.

Lemma 4.8. In every execution of $\mathcal{M}_e$ on a graph $\mathbf{G}$ such that $Dir(\mathbf{G})$ is nt-minimal, exactly one vertex v is declared as *elected*.

Proof:
One knows that every maximal execution of $\mathcal{M}_e$ terminates. First, suppose that after the final step i, there exists a vertex v such that $a_i(v) \leq |V(G)|$. Since all messages have been heard, for every $v \in V(G)$, for every $(n,p) \in N(v)$, $p > 0$ and for every $(n,p,a) \in A(v)$, $p > 0$. Among all vertices v such that $a_i(v)$ is minimal, let v be the last one that hears a message and let i_0 be the step where v hears this last message. After v has processed the message $a_{i_0}(v) = 1 + \min\{a \mid \exists (n,p,a) \in A(v)\}$. Thus, there exists a neighbour w of v such that $a_i(w) = a_{i_0}(v) - 1 = a_i(v) - 1$, which is a contradiction with our choice of v.

From Lemmas 4.6 and 4.7, if there exist a step i_0 and a vertex $v \in V(G)$ such that $a_{i_0}(v) > |V(G)|$, then $\mathbf{H}(M(v), i_0)$ and $Dir(\mathbf{G})$ are isomorphic. Moreover, from Lemma 4.5, we know that all the vertices have the same mailbox and that for each w, $n(w), N(w)$ and $M(w)$ will not change anymore. Consequently, after step i_0, for any w, the digraph $\mathbf{D}(M(w))$ is always $\mathbf{D}(M_{i_0}(v))$. Thus, there exists a step i such that for all $w \in V(G)$, $M_i(w) = M_{i_0}(v)$ and $a_i(w) > |V(G)|$. Let $M = M_{i_0}(v)$. Since $Dir(\mathbf{G})$ is t-minimal, $C_{\mathbf{G},\mathbf{D}(M)}$ is not empty. Thus, there is a unique vertex $v \in V(G)$ such that $n_i(v) = \min C_{\mathbf{G},\mathbf{D}(M)}$, and this vertex is elected. □

Therefore, we have proven the following theorem:

Theorem 4.9. For every graph $\mathbf{G}$, there exists a polynomial (memory, messages and size of messages) leader election algorithm on $\mathbf{G}$ using asynchronous broadcast communications if and only if the digraph $Dir(\mathbf{G})$ is nt-minimal.

4.6. Remarks on the Initial Knowledge: Degree Awarness

From previous assumptions on the initial knowledge, an interesting question could be to know what happens when processes initially know their degree.

Let $\mathbf{G}$ be a labelled digraph such that $Dir(\mathbf{G})$ is nt-minimal. If each process knows its degree and the size of the graph, one can modify the algorithm $\mathcal{M}_e$ (Algorithm 2) to take into account this combination of knowledge. Before increasing the confidence level in which all processes have the same mailbox, each process v waits until it has received a message from all its neighbouring processes. Once the sum of p such that $(n,p) \in N(v)$ is equal to the degree $deg(v)$ of v, we deduce that v has received a message from all of its neighbouring processes at least once. From Lemma 4.5, for each step i, the ball in $\mathbf{G}$ centered at v of radius $a_i(v)$ belongs to $H(M,i)$. Hence, if $a_i(v) > |V(G)|$ then $\mathbf{H}(M, i(v))$ and $Dir(\mathbf{G})$ are isomorphic. Note that knowing the diameter of the graph is sufficient. The radius of the ball centered at v only increases when $a_i(v) \leq a_i(w)$ for every $w \in N_{\mathbf{G}}(v)$. Consequently, let $Diam(G)$ be the diameter of $\mathbf{G}$, if $a_i(v) > Diam(G)$, we can easily extend our proofs and deduce that $\mathbf{H}(M, i(v))$ and $Dir(\mathbf{G})$ are isomorphic.

We previously showed (Lemmas 4.7 and 4.6) that once each process has a confidence level greater than the size of the graph, then all processes have the same mailbox and are able to reconstruct the same digraph $\mathbf{D}$. We also stated (Proposition 3.7) that the digraph $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$. The following lemma establishes a link between the degree of each process and the size of its fibre:

Lemma 4.10. ([5])
Let $\mathbf{D}$ be a labelled digraph, we denote $d_{(v,v')}$ (resp. $d_{(v',v)}$), the number of arcs a such that $s(a) = v$ and $t(a) = v'$ (resp. $s(a) = v'$ and $t(a) = v$) in $\mathbf{D}$. For every pair of vertices $v, v' \in V(D)$, there exist two integers $d_{(v,v')}, d_{(v',v)}$ such that given a simple graph $\mathbf{G}$, if $Dir(\mathbf{G})$ is fibred over $\mathbf{D}$ via φ, then $d_{(v,v')}|\varphi^{-1}(v)| = d_{(v',v)}|\varphi^{-1}(v')|$.

With the initial knowledge of its degree, a process can compute from Lemma 4.10 the size of the fibre of each process that belongs to the digraph $\mathbf{D}(M(v))$ reconstructed from its mailbox $M(v)$. Thus, every process can locally identify processes that belong to the set of candidates (Definition 2.10) of the reconstructed graph $\mathbf{D}$. Therefore, the elected process is the vertex with the smallest identity of this set. Hence, our leader election algorithm can be easily used in the model in which each process is endowed with degree-awarness (see [5]) while keeping a polynomial complexity and asynchronous broadcast communications.

Remark 4.11. From Lemma 4.10, given a minimal digraph $\mathbf{B}$, we know that for any simple graph $\mathbf{G}$ that is fibred over $\mathbf{B}$, the set of candidates $C_{\mathbf{G},\mathbf{B}}$ does not depend on $\mathbf{G}$, but only on $\mathbf{B}$.

In Algorithm 2, since processes only use the minimal base $\mathbf{B}_{\mathbf{G}}$ of $Dir(\mathbf{G})$, one can relax the initial knowledge of every process. In order to solve the leader election problem in our model, it suffices that each process knows the size of the graph and the minimal base $\mathbf{B}_{\mathbf{G}}$ — and not necessarily the initial graph $\mathbf{G}$.

References

[1] Angluin, D.: Local and global properties in networks of processors, *Proc. of the 12th Symposium on Theory of Computing (STOC 1980)*, 1980, 82–93.

[2] Angluin, D., Aspnes, J., Eisenstat, D., Ruppert, E.: The computational power of population protocols, *Distributed Computing*, **20**(4), November 2007, 279–304.

[3] Bang-Jensen, J., Gutin, G.: *Digraphs - Theory, Algorithms and Applications,* Springer, 2002, ISBN 978-1-85233-611-0.

[4] Bodlaender, H.: The classification of coverings of processor networks, *Journal of parallel and distributed computing*, **6**(1), 1989, 166–182.

[5] Boldi, P., Codenotti, B., Gemmell, P., Shammah, S., Simon, J., Vigna, S.: Symmetry breaking in anonymous networks: characterizations, *Proc. of the 4th Israeli Symposium on Theory of Computing and Systems (ISTCS 1996),* IEEE Press, 1996, 16–26.

[6] Boldi, P., Vigna, S.: Fibrations of graphs, *Discrete Mathematics*, **243**(1-3), 2002, 21–66.

[7] Chalopin, J., Métivier, Y.: An Efficient Message Passing Election Algorithm based on Mazurkiewicz's Algorithm, *Fundamenta Informaticae*, **80**(1-3), 2007, 221–246.

[8] Chalopin, J., Métivier, Y.: On the power of synchronization between two adjacent processes, *Distributed Computing*, **23**(3), 2010, 177–196.

[9] Chalopin, J., Métivier, Y., Zielonka, W.: Local computations in graphs: the case of cellular edge local computations, *Fundamenta Informaticae*, **74**(1), 2006, 85–114.

[10] Chlebus, B.: Randomized Communication in Radio Networks, in: *Handbook of Randomized Computing* (P. Pardalos, S. Rajasekaran, J. Reif, J. Rolim, Eds.), vol. I, Kluwer Academic Publishers, 2001, 401–456,.

[11] Cidon, I., Mokryn, O.: Propagation and Leader Election in a Multihop Broadcast Environment, *DISC*, 1998, 104–118.

[12] Godard, E., Métivier, Y., Muscholl, A.: Characterization of classes of graphs recognizable by local computations, *Theory of Computing Systems*, **37**(2), 2004, 249–293.

[13] Guerraoui, R., Ruppert, E.: What Can Be Implemented Anonymously?, *DISC*, 2005, 244–259.

[14] Helary, J.-M., Raynal, M.: Assigning distinct identities to sites on an anonymous distributed system, *Distributed Computing Systems in the 1990s, 1988. Proceedings., Workshop on the Future Trends of*, sep 1988, 82 –86.

[15] LeLann, G.: Distributed systems, towards a formal approach, *Information processing 1977*, North-Holland, 1977, 155–160.

[16] Massey, W.: *A basic course in algebraic topology*, Springer-Verlag, 1991, Graduate texts in mathematics.

[17] Mazurkiewicz, A.: Distributed enumeration, *Information Processing Letters*, **61**(5), 1997, 233–239.

[18] Santoro, N.: *Design and Analysis of Distributed Algorithms (Wiley Series on Parallel and Distributed Computing)*, Wiley-Interscience, 2006, ISBN 0471719978.

[19] Stojmenovic, I.: *Handbook of Sensor Networks : Algorithms and Architectures*, Wiley, 2005.

[20] Szymanski, B., Shy, Y., Prywes, N.: Terminating iterative solutions of simultaneous equations in distributed message passing systems, *Proc. of the 4th Annual ACM Symposium on Principles of Distributed Computing (PODC 1985)*, ACM Press, 1985, 287–292.

[21] Tanenbaum, A., van Steen, M.: *Distributed Systems - Principles and Paradigms*, Prentice Hall, 2002.

[22] Tel, G.: *Introduction to distributed algorithms*, Cambridge University Press, 2000.

[23] Yamashita, M., Kameda, T.: Computing on anonymous networks: Part I - Characterizing the solvable cases, *IEEE Transactions on parallel and distributed systems*, **7**(1), 1996, 69–89.

[24] Yamashita, M., Kameda, T.: Computing on anonymous networks: Part II - decision and membership problems, *IEEE Transactions on parallel and distributed systems*, **7**(1), 1996, 90–96.

[25] Yamashita, M., Kameda, T.: Leader election problem on networks in which processor identity numbers are not distinct, *IEEE Transactions on parallel and distributed systems*, **10**(9), 1999, 878–887.